Class 9: Structural Bioinformatics

Soomin Park

Protien Data Bank (PDB) Statistics

The PDB is the main database for structural information on biomolecules. Let's see what it contains:

```
db <- read.csv("Data Export Summary.csv")
db
```

```
          Molecular.Type   X.ray     EM    NMR Multiple.methods Neutron Other
1          Protein (only) 154,766 10,155 12,187              191      72    32
2 Protein/Oligosaccharide   9,083  1,802     32                7       1     0
3              Protein/NA   8,110  3,176    283                6       0     0
4     Nucleic acid (only)   2,664     94  1,450               12       2     1
5                   Other     163      9     32                0       0     0
6  Oligosaccharide (only)      11      0      6                1       0     4
    Total
1 177,403
2  10,925
3  11,575
4   4,223
5     204
6      22
```

Q1: What percentage of structures in the PDB are solved by X-Ray and Electron Microscopy.

```
# remove commas
gsub("," , "", db$X.ray)
```

```
[1] "154766" "9083"   "8110"   "2664"   "163"    "11"
```

```
# convert character to numeric
as.numeric(gsub("," , "", db$X.ray))
```

```
[1] 154766   9083   8110   2664    163     11
```

```
# find the sum
sum(as.numeric(gsub("," , "", db$X.ray)))
```

```
[1] 174797
```

```
xray.total <- sum(as.numeric(gsub("," , "", db$X.ray)))
print(xray.total)
```

```
[1] 174797
```

```
em.total <- sum(as.numeric(gsub("," , "", db$EM)))
print(em.total)
```

```
[1] 15236
```

Hmm… I am doing the same thing over and over. Time to write a function.

```
# I will work with 'x' as input and make a working snippet.

x <- db$X.ray
sum(as.numeric(gsub("," , "", x)))
```

```
[1] 174797
```

```
# Code > Extract Function

sum_comma <- function(x) {
  # Substitute the comma and convert to numeric
  sum(as.numeric(gsub("," , "", x)))
}
```

```
# Test the function
sum_comma(db$X.ray)
```

```
[1] 174797
```

```
sum_comma(db$Total)
```

```
[1] 204352
```

Now we can use the function to find the percentage of structures solved by X-ray

```
sum_comma(db$X.ray) / sum_comma(db$Total) * 100
```

```
[1] 85.53721
```

For Electron Microscopy:

```
sum_comma(db$EM) / sum_comma(db$Total) * 100
```

```
[1] 7.455763
```

Q2: What proportion of structures in the PDB are protein?

```
# find the total number of protein and remove comma. (There are two ways)
sum_comma(db[1, 8])
```

```
[1] 177403
```

```
sum_comma(db$Total[1])
```

```
[1] 177403
```

```
sum_comma(db$Total[1]) / sum_comma(db$Total)
```

```
[1] 0.8681246
```

```
# round  the number
round(sum_comma(db$Total[1]) / sum_comma(db$Total), 2)
```

```
[1] 0.87
```

Q3: Type HIV in the PDB website search box on the home page and determine how many HIV-1 protease structures are in the current PDB?

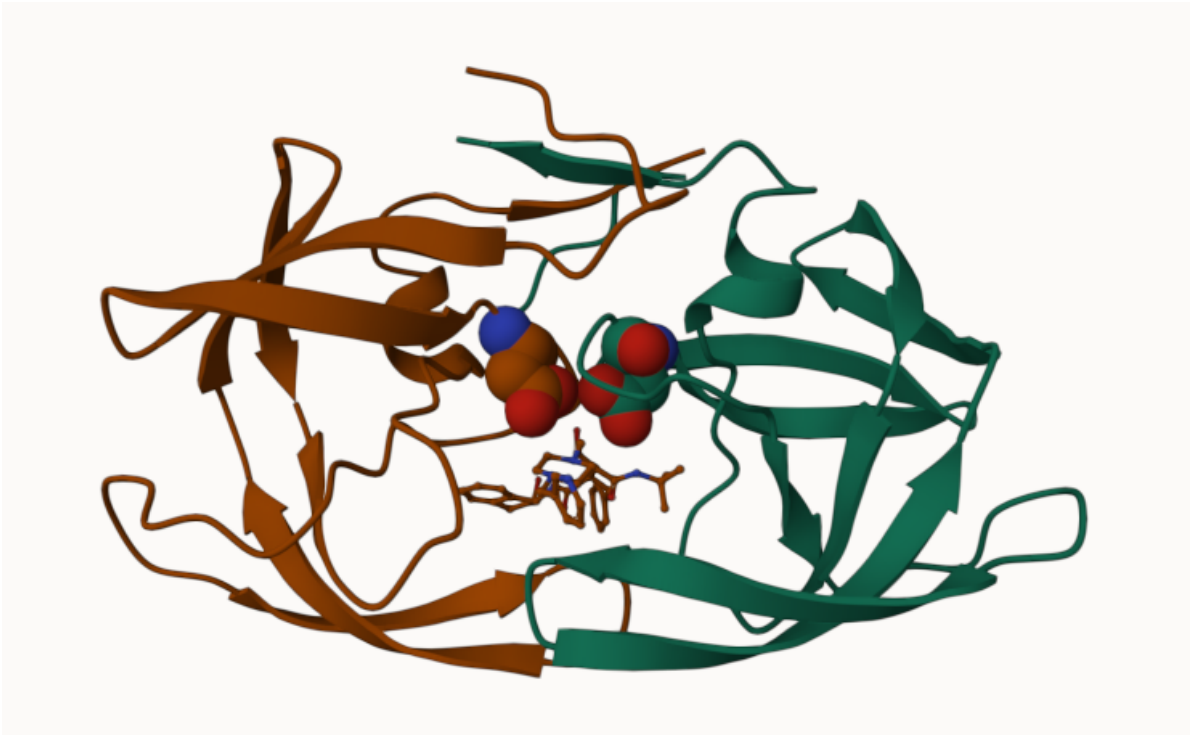


Figure 1: HIV-PR structure from MERK with a bound drug

Visualizing the HIV-1 protease structure

Q4: Water molecules normally have 3 atoms. Why do we see just one atom per water molecule in this structure?

The structure is too low a resolution to see H atoms. You need a sub 1 Angstrom resolution to see Hydrogen.

Q5: There is a critical "conserved" water molecule in the binding site. Can you identify this water molecule? What residue number does this water molecule have

HOH308

Working with Structures in R (Introduction to Bio3D in R)

We can use the `bio3d` package to read and perform bioinformatics calculations on PDB structures.

```
library(bio3d)

pdb <- read.pdb("1hsg")
```

```
 Note: Accessing on-line PDB file
```

```
pdb
```

```
 Call:  read.pdb(file = "1hsg")

   Total Models#: 1
     Total Atoms#: 1686,  XYZs#: 5058  Chains#: 2  (values: A B)

     Protein Atoms#: 1514  (residues/Calpha atoms#: 198)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 172  (residues: 128)
     Non-protein/nucleic resid values: [ HOH (127), MK1 (1) ]

   Protein sequence:
      PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYD
      QILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFPQITLWQRPLVTIKIGGQLKE
      ALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYDQILIEICGHKAIGTVLVGPTP
      VNIIGRNLLTQIGCTLNF

+ attr: atom, xyz, seqres, helix, sheet,
         calpha, remark, call
```

Q7: How many amino acid residues are there in this pdb object?

198

Q8: Name one of the two non-protein residues?

MK1

Q9: How many protein chains are in this structure?

2

```
attributes(pdb)
```

```
$names
[1] "atom"   "xyz"    "seqres" "helix"  "sheet"  "calpha" "remark" "call"

$class
[1] "pdb" "sse"
```

```
head(pdb$atom)
```

```
  type eleno elety  alt resid chain resno insert      x      y     z o     b
1 ATOM     1     N <NA>   PRO     A     1   <NA> 29.361 39.686 5.862 1 38.10
2 ATOM     2    CA <NA>   PRO     A     1   <NA> 30.307 38.663 5.319 1 40.62
3 ATOM     3     C <NA>   PRO     A     1   <NA> 29.760 38.071 4.022 1 42.64
4 ATOM     4     O <NA>   PRO     A     1   <NA> 28.600 38.302 3.676 1 43.40
5 ATOM     5    CB <NA>   PRO     A     1   <NA> 30.508 37.541 6.342 1 37.87
6 ATOM     6    CG <NA>   PRO     A     1   <NA> 29.296 37.591 7.162 1 38.40
  segid elesy charge
1  <NA>     N   <NA>
2  <NA>     C   <NA>
3  <NA>     C   <NA>
4  <NA>     O   <NA>
5  <NA>     C   <NA>
6  <NA>     C   <NA>
```

```
adk <- read.pdb("6s36")
```

```
  Note: Accessing on-line PDB file
   PDB has ALT records, taking A only, rm.alt=TRUE
```

```
adk
```

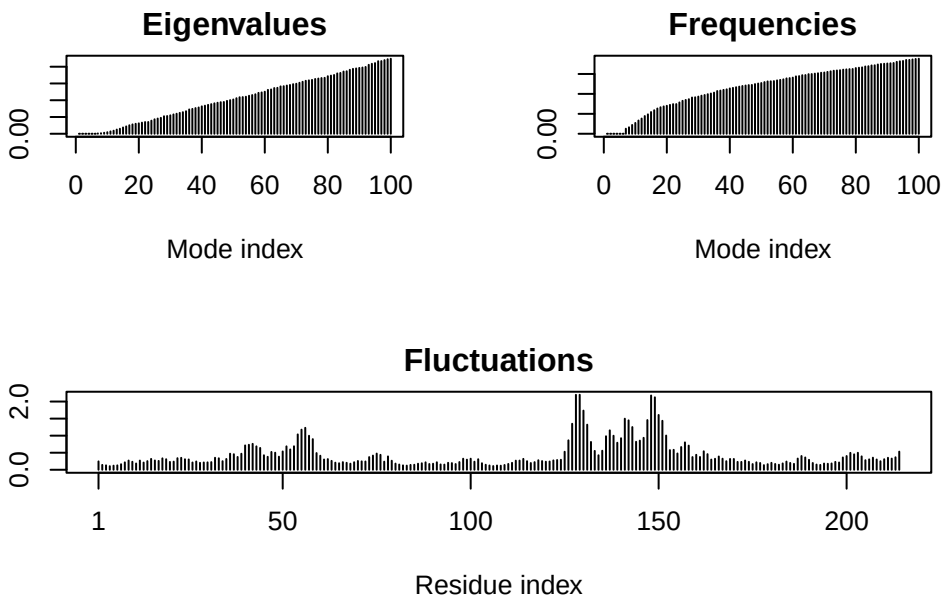
```
 Call:  read.pdb(file = "6s36")

   Total Models#: 1
     Total Atoms#: 1898,  XYZs#: 5694  Chains#: 1  (values: A)

     Protein Atoms#: 1654  (residues/Calpha atoms#: 214)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 244  (residues: 244)
     Non-protein/nucleic resid values: [ CL (3), HOH (238), MG (2), NA (1) ]

   Protein sequence:
      MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
      DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDKI
      VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
      YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG

+ attr: atom, xyz, seqres, helix, sheet,
        calpha, remark, call
```

Perform a prediction of flexibility with a technique called NMA (normal mode analysis)

```
# Perform flexiblity prediction
m <- nma(adk)
```

```
 Building Hessian...        Done in 0.053 seconds.
 Diagonalizing Hessian...   Done in 0.461 seconds.
```

```
plot(m)
```



Write out a "movie" of the motion for viewing in MolStar

```
mktrj(m, file="adk_m7.pdb")
```